推理系统系列

推理系统架构

ZOMI

Talk Overview

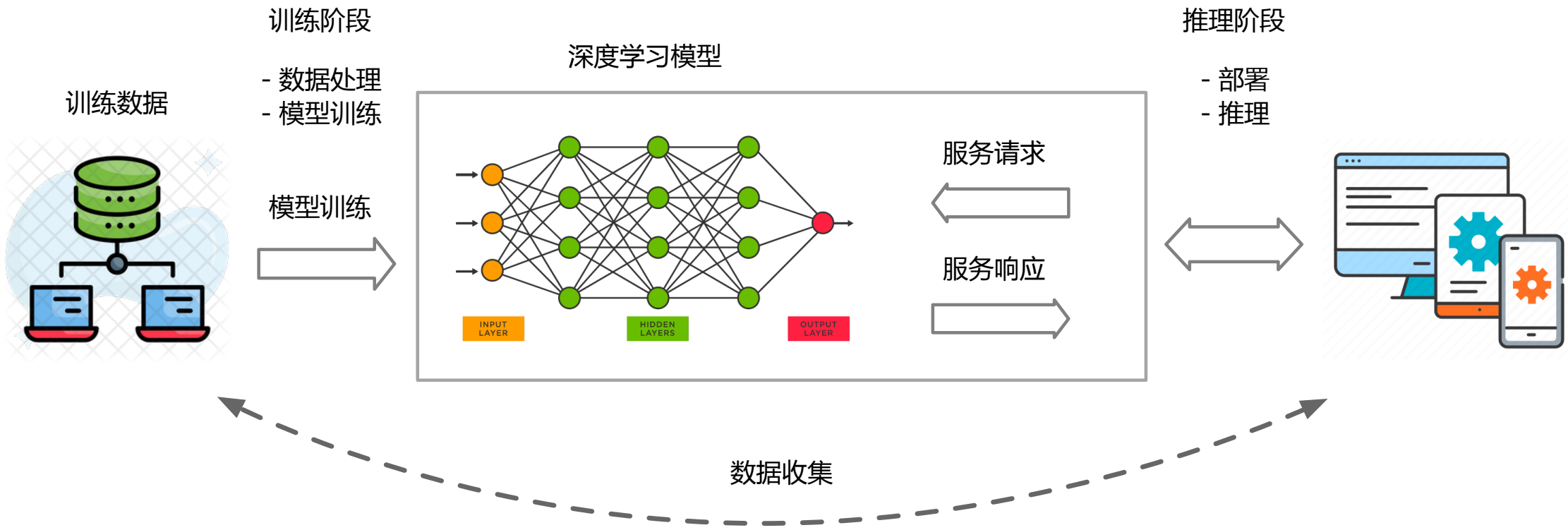
1. **推理系统介绍**
   - 推理系统与推理引擎区别
   - 推理工作流程
   - 推理系统架构
   - 推理引擎架构
2. **模型小型化**
   - NAS神经网络搜索
   - CNN小型化结构
   - Transform小型化结构
3. **离线优化压缩**
   - 低比特量化
   - 二值化网络
   - 模型模型剪枝
   - 模型模型蒸馏
4. **部署和运行优化**
   - 图转换优化（算子融合/重排/替换）
   - 并发执行与内存分配
   - 动态batch与bin Packing

Talk Overview

I. 推理系统架构

- Infer, Deploy, Serve – 推理、部署、服务化
- Architecture of Inference System – 推理系统的整体架构
- Life Time of Model – 模型生命周期管理

深度学习模型的生命周期



深度学习模型的生命周期

- **训练任务**：数据中心中更像是传统的批处理任务，需要执行数小时，数天才能完成，其一般配置较大的批尺寸追求较大的吞吐，将模型训练达到指定的准确度或错误率。
- **推理任务**：执行 7 X 24 的服务，其常常受到响应延迟的约束，配置的批尺寸更小，模型已经稳定一般不再被训练。

AI 部署 Question?

- AI 应用部署需要考虑哪些方面？

推理、部署、服务化

Question?

1. 什么是模型推理？什么是推理服务化？
2. 有哪些常见的推理服务框架？Triton是什么？

概念澄清

推理（Inference）

- 对于训练（Training）而言的推理，即模型前向计算，也就是对于给出的输入数据计算得到模型的输出结果；相对预测（Prediction）的推理，是统计学领域的范畴。

部署（Deployment）

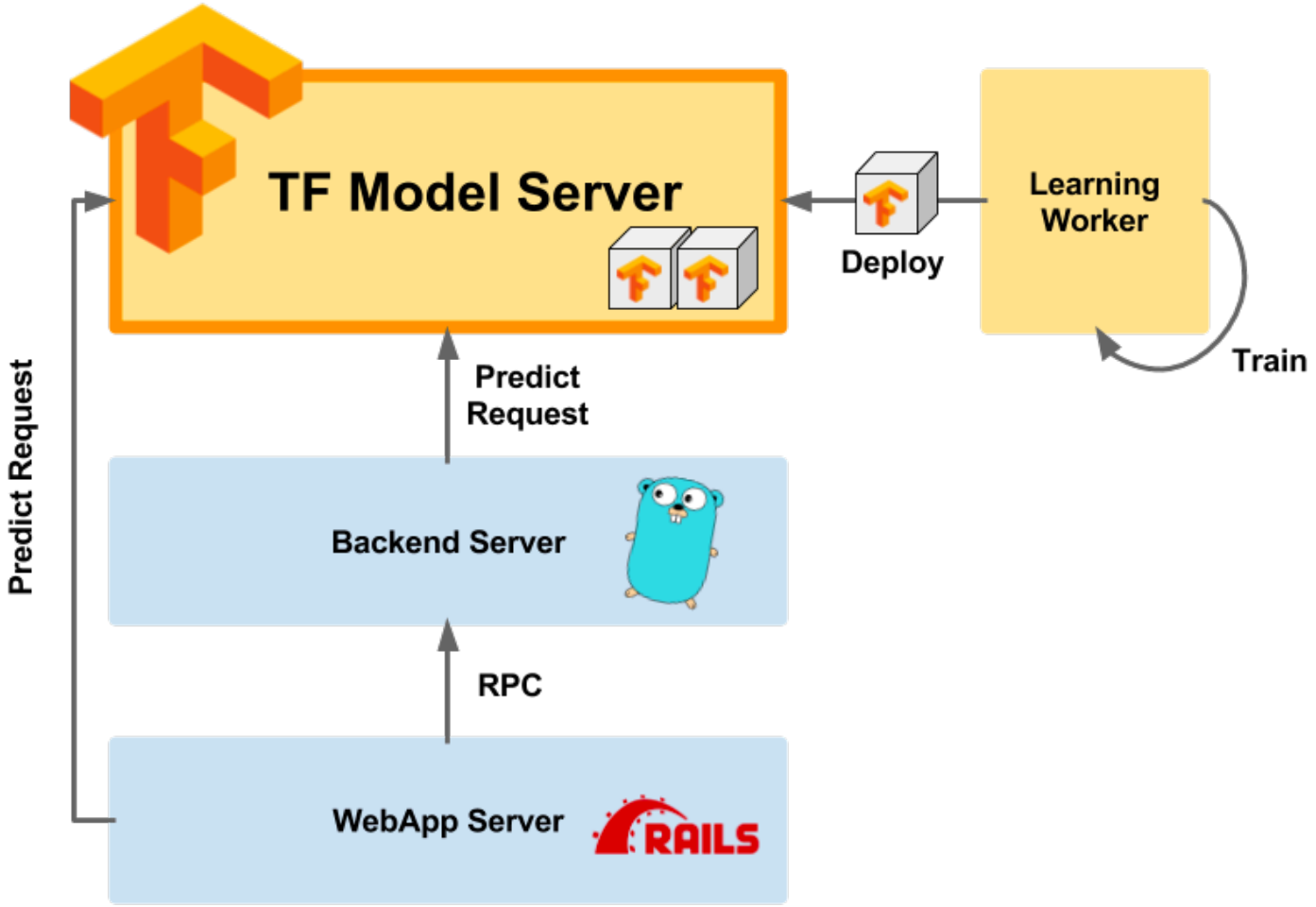
- 训练得到的模型主要目的还是为了更有效地解决实际中的问题，因此部署是一个非常重要的阶段。模型部署的课题也非常多，包括但不仅限于：移植、压缩、加速等。

服务化（Serving）

- 模型的部署方式是多样的：封装成一个SDK，集成到APP或者服务中；封装成一个web服务，对外暴露接口（HTTP(S)，RPC等协议）。

TF Serving

- Google 早在 2016 年就针对 TensorFlow 推出了服务化框架 TensorFlow Serving，能够把 TensorFlow 模型以 web 服务的方式对外暴露接口，通过网络请求方式接受来自客户端（Client）的请求数据，计算得到前向推理结果并返回。

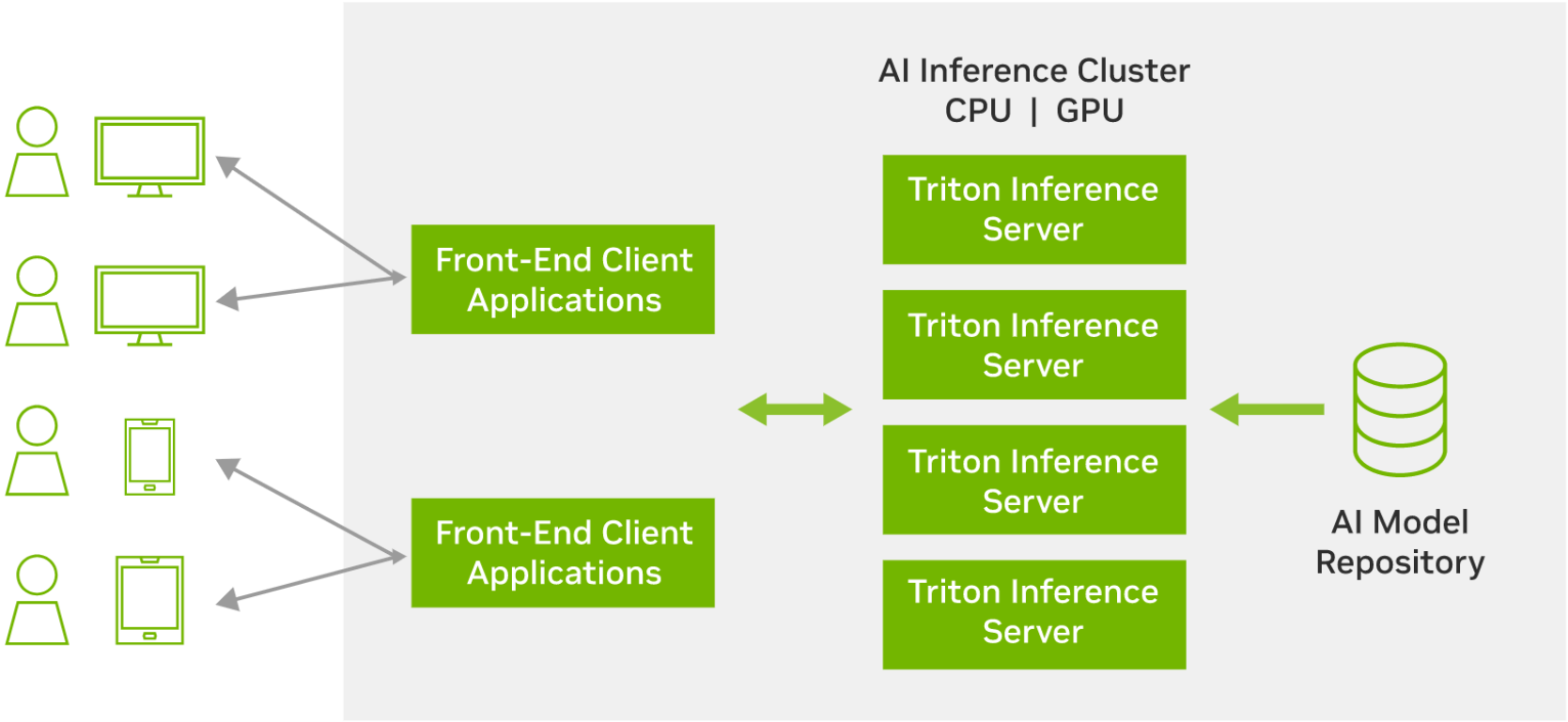


常见的服务化框架

服务框架	支持的模型	开源仓库地址	开源时间
TensorFlow Serving	TensorFlow	https://github.com/tensorflow/serving	2016
TorchServe	PyTorch	https://github.com/pytorch/serve	2020
Triton	TensorFlow/PyTorch等	https://github.com/triton-inference-server/server	2018
BentoML	TensorFlow/PyTorch等	https://github.com/bentoml/BentoML	2019
Kubeflow	TensorFlow/PyTorch等	https://github.com/kubeflow/kfserving	2019
Seldon Core	TensorFlow/PyTorch等	https://github.com/SeldonIO/seldon-core	2018

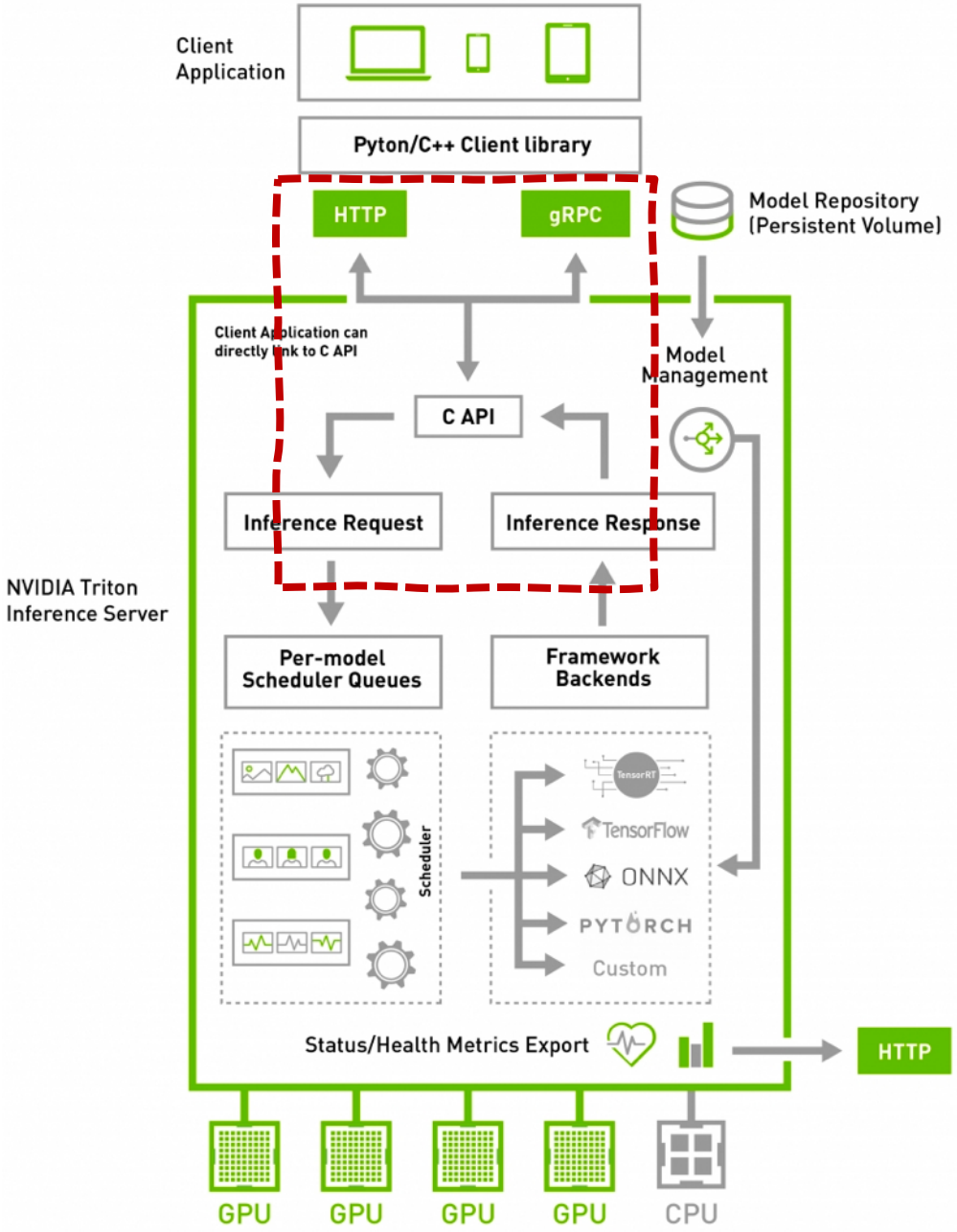
Triton

- Triton推理服务器 (NVIDIA Triton Inference Server) 是英伟达等公司推出的开源推理框架，为用户提供部署在云和边缘推理上的解决方案。



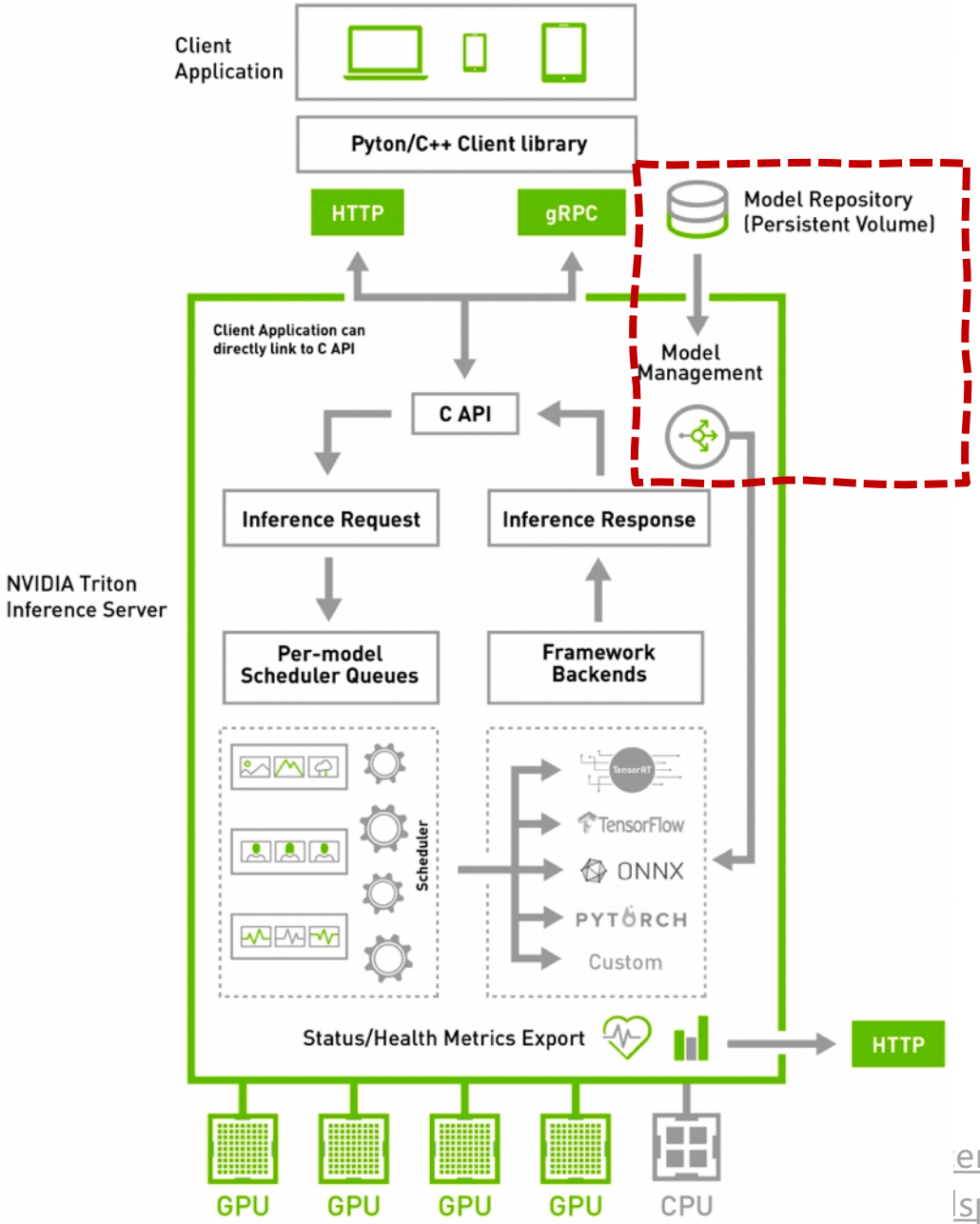
Triton 接入层

- Triton 支持 HTTP/REST 和 GRPC 协议。其实除此之外，Triton 还支持共享内存（Shared Memory）的 IPC（Inter-Process Communication）通信机制。



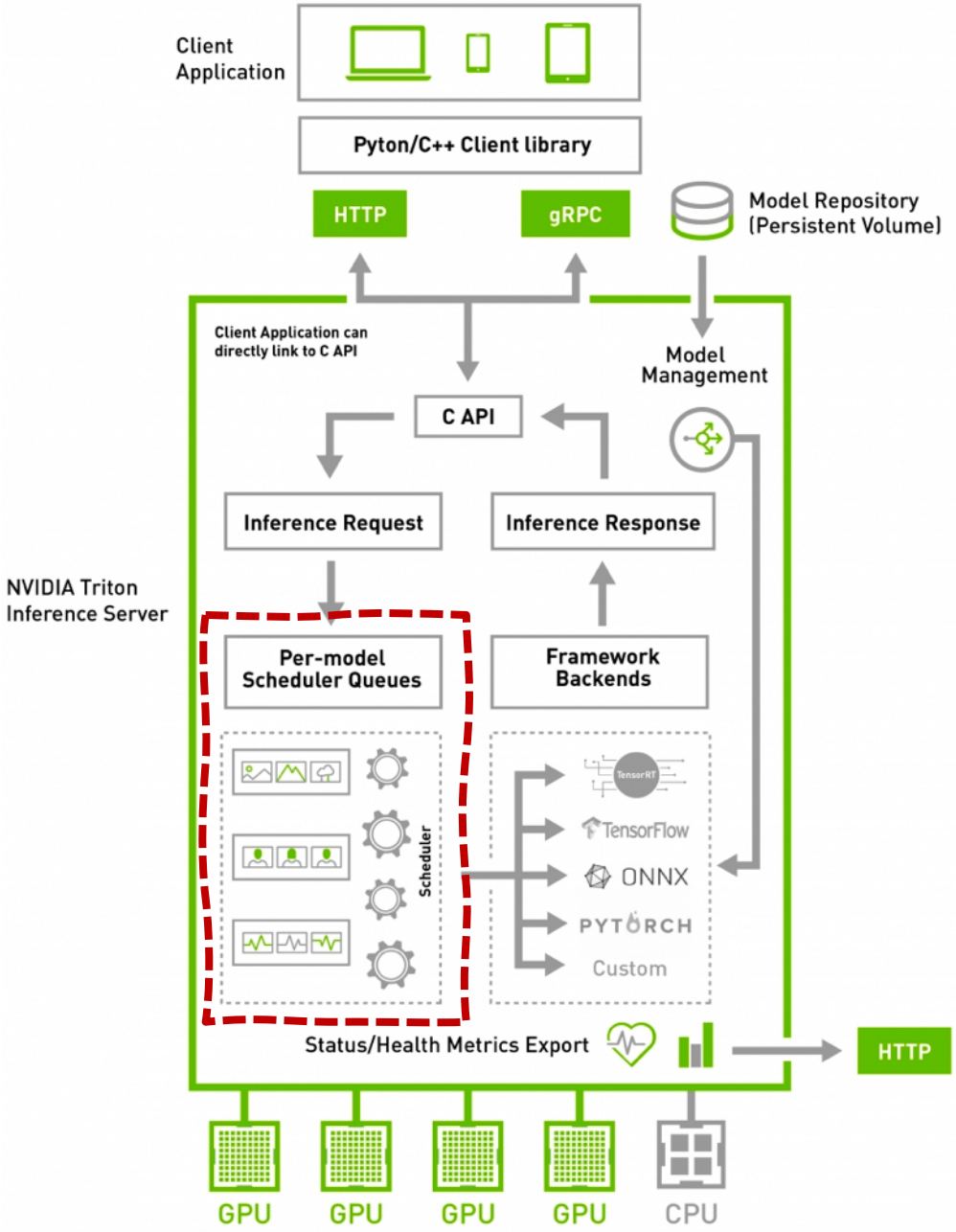
Triton 模型仓库

- 模型仓库可以是本地的持久化存储介质（磁盘），也可以接入 Google Cloud Platform 或者 AWS S3 模型仓库。Triton 的模型仓库支持多模型、也支持模型编排。



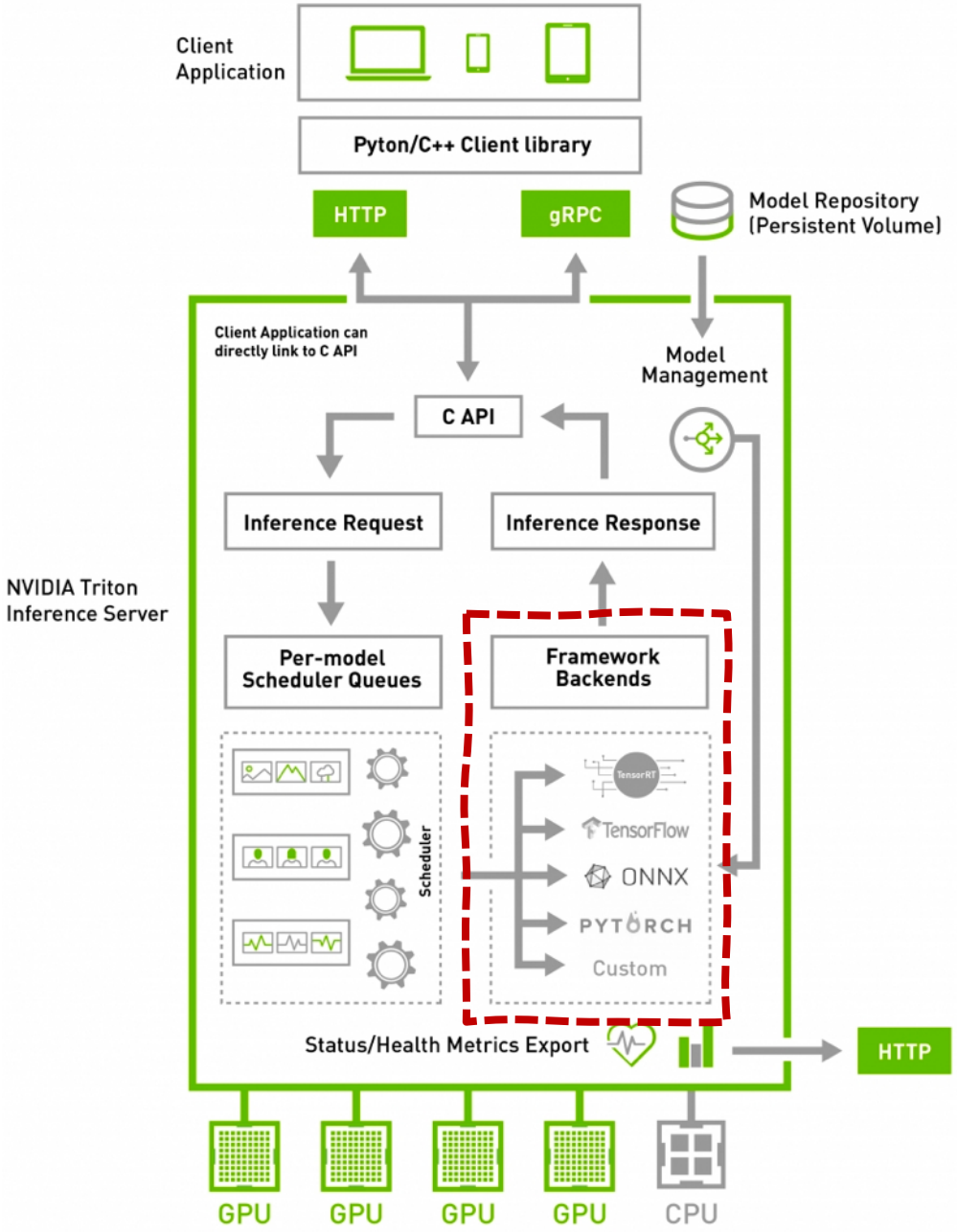
Triton 模型预编排

- Pre-Model Scheduler Queues，核心工作是模型编排：通过解析请求的URL，从模型仓库查询到编排信息，执行模型编排。



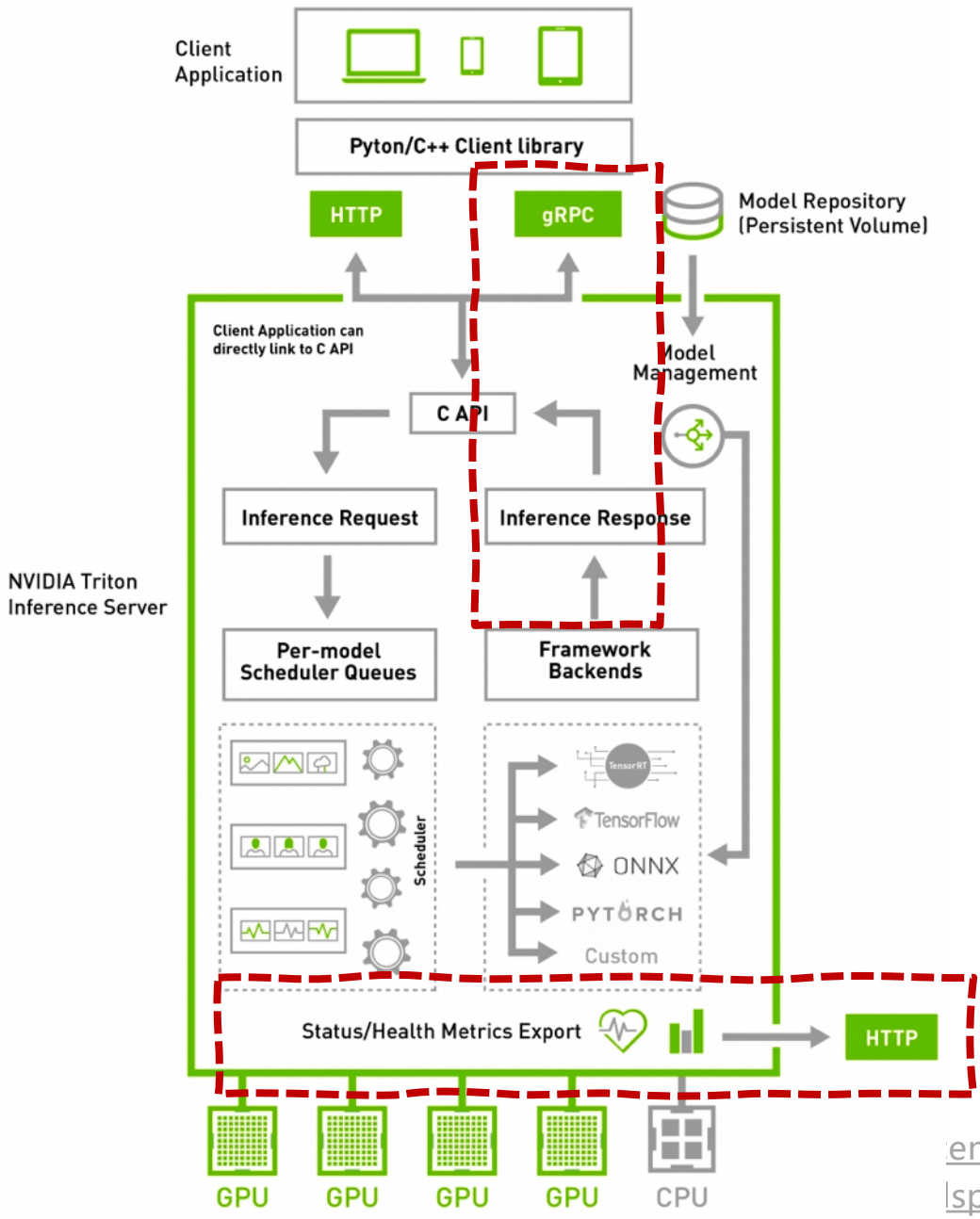
Triton 推理引擎

- Triton 支持 TensorFlow, TensorRT, PyTorch, ONNX Runtime 推理引擎，Triton统一称为“Backend”。在 Triton 开始启动时，模型仓库中的模型就已经被加载到内存或者显存上了，然后服务调起推理引擎执行实际的计算。



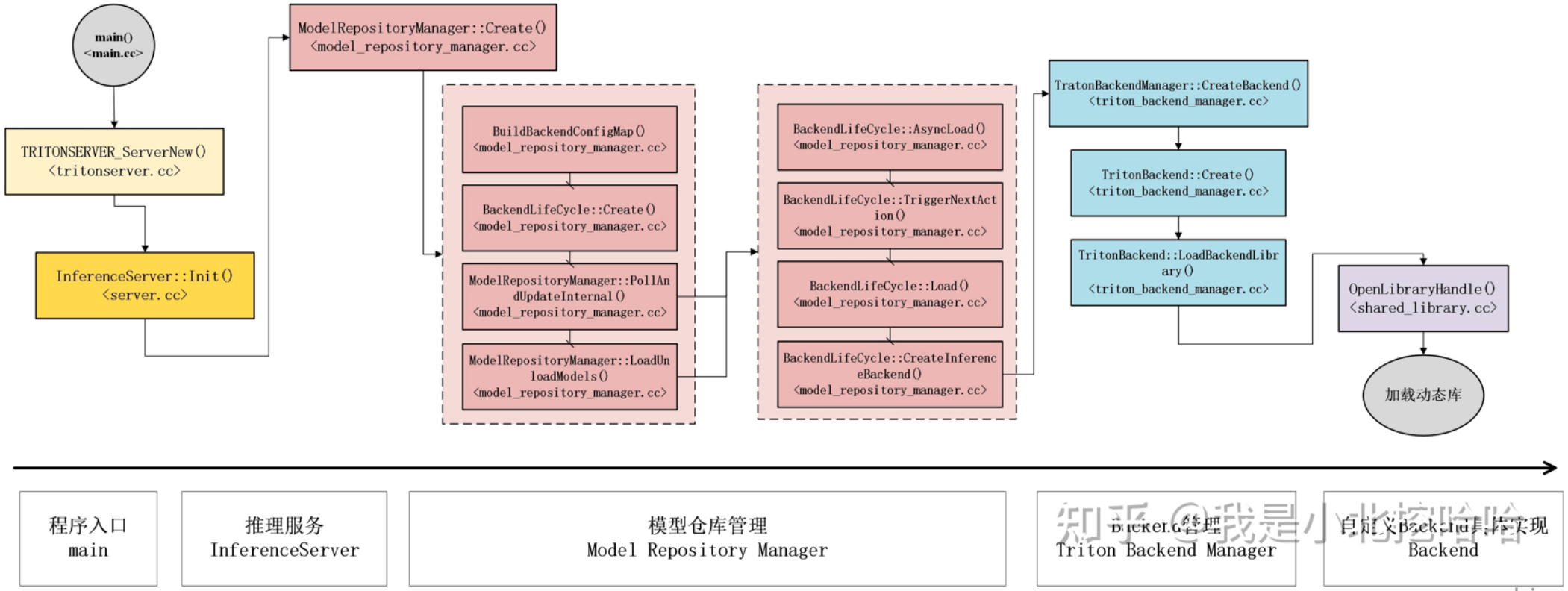
Triton 返回与监控

- Inference Response 为结果返回，即把最终结果返回给客户端。
- Status/Health Metrics Export 是 Triton 支持接入 Prometheus 监控的接口。



基于 Triton 集成推理引擎

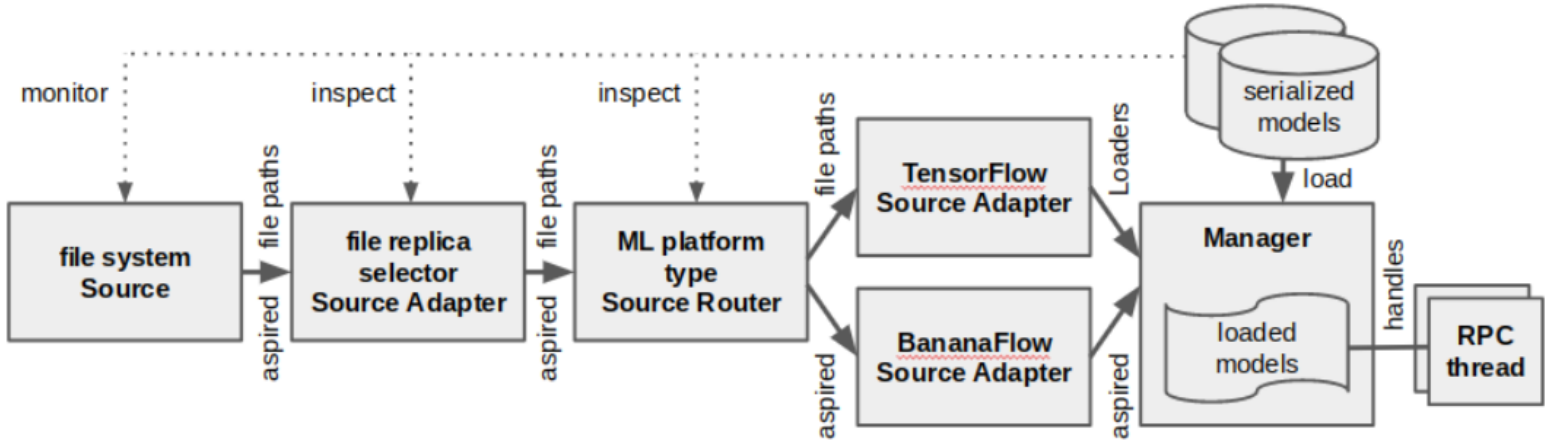
- 网络请求和模型编排等相关的功能，Triton服务已经集成好了，Backend只需要关心模型的加载（Load）、前向推理计算（Forward）和卸载（Unload），以及配置文件校验。



模型生命周期管理

模型版本管理

- **需要模型版本管理的原因**
  - 每隔一段时间训练出的新版本模型替换线上模型，但是可能存在缺陷
  - 如果新版本模型发现缺陷需要回滚
- **模型生命周期管理**
  - 金丝雀(Canary)策略
  - 回滚(Rollback)策略



模型生命周期管理工作流实例

TensorFlow Serving: Flexible, High-Performance ML Serving

金丝雀 Canary 策略

金丝雀策略

- 当获得一个新模型版本，当前服务模型成为 second-newest，用户可以选择同时保持这两个版本
- 将所有推理请求流量发送到当前两个版本，比较它们的效果
- 一旦对最新版本达标，用户就可以切换到仅该版本
- 方法需要更多的高峰资源，避免将用户暴露于缺陷模型

TensorFlow Serving: Flexible, High-Performance ML Serving

回滚 Rollback 策略

回滚策略

- 如果在当前的主要服务版本上检测到缺陷，则用户可以请求切换到特定的较旧版本
- 卸载和装载的顺序应该是可配置的
- 当问题解决并且获取到新的安全版本模型时，从而结束回滚

TensorFlow Serving: Flexible, High-Performance ML Serving

参考文献

1. Deep Learning Inference in Facebook Data Centers: Characterization, Performance Optimizations and Hardware Implications
2. Clipper: A Low-Latency Online Prediction Serving System
3. TFX: A TensorFlow-Based Production-Scale Machine Learning Platform
4. TensorFlow-Serving: Flexible, High-Performance ML Serving
5. Optimal Aggregation Policy for Reducing Tail Latency of Web Search
6. A Survey of Model Compression and Acceleration for Deep Neural Networks
7. CSE 599W: System for ML - Model Serving
8. https://developer.nvidia.com/deep-learning-performance-training-inference
9. DEEP COMPRESSION: COMPRESSING DEEP NEURAL NETWORKS WITH PRUNING, TRAINED QUANTIZATION AND HUFFMAN CODING
10. Learning both Weights and Connections for Efficient Neural Networks
11. DEEP LEARNING DEPLOYMENT WITH NVIDIA TENSORRT
12. Halide: A Language and Compiler for Optimizing Parallelism,Locality, and Recomputation in Image Processing Pipelines
13. TVM: An Automated End-to-End Optimizing Compiler for Deep Learning
14. 8-bit Inference with TensorRT
15. https://github.com/microsoft/AI-System
16. 模型推理服务化之Triton：如何基于Triton开发自己的推理引擎？ https://zhuanlan.zhihu.com/p/354058294